

AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1 1. (Previously presented) A method for determining a class dependency
2 that identifies a supporting class on which a target class depends, wherein the
3 target class is defined in an object-oriented programming language, comprising:
4 receiving a representation of the target class at a first platform-independent
5 virtual machine;
6 creating a model of the target class from the representation;
7 analyzing the model to detect references to the supporting class;
8 if a supporting class is detected, determining a class dependency for the
9 supporting class;
10 creating a list of dependent classes for the target class and supporting
11 classes; and
12 sharing the list of dependent classes with a second platform-independent
13 virtual machine so that the second platform-independent virtual machine does not
14 need to create the list of dependent classes.

1 2. (Original) The method of claim 1, further comprising, identifying
2 classes that an object depends upon by:
3 receiving a representation of the object;
4 serializing the referenced object;
5 parsing the data resulting from the object serialization to identify classes
6 referenced from the target object's properties, configuration, or state; and

7 determining the dependent classes of the referenced object.

1 3. (Original) The method of claim 1, further comprising saving the list of
2 dependent classes to a storage structure.

1 4. (Original) The method of claim 3, wherein the storage structure is one
2 of a hash table and a database.

1 5. (Original) The method of claim 1, wherein creating the list of dependent
2 classes includes creating one of a distribution list and a distribution file.

1 6. (Original) The method of claim 2, further comprising:
2 inserting the object into an object database;
3 determining if the target class and supporting classes for the target class
4 are in the class path; and
5 adding the target class and supporting classes for the target class to the
6 class path if necessary.

1 7. (Original) The method of claim 2, further comprising:
2 retrieving the object from the object database;
3 determining if the target class and supporting classes for the target class
4 are in the class path; and
5 adding the target class and supporting classes for the target class to the
6 class path if necessary.

1 8. (Original) The method of claim 1, further comprising filtering the list of
2 identified classes to remove duplicate and core class references.

1 9. (Original) The method of claim 1, further comprising saving the list of
2 dependent classes of the target class as well as the list of dependent classes of the
3 supporting classes in cache to facilitate subsequent lookups of dependent classes
4 of the target class.

1 10. (Previously presented) A computer-readable storage medium storing
2 instructions that when executed by a computer cause the computer to perform a
3 method for determining a class dependency that identifies a supporting class on
4 which a target class depends, wherein the target class is defined in an object-
5 oriented programming language, the method comprising:
6 receiving a representation of the target class at a first platform-independent
7 virtual machine;
8 creating a model of the target class from the representation;
9 analyzing the model to detect references to the supporting class;
10 if a supporting class is detected, determining a class dependency for the
11 supporting class;
12 creating a list of dependent classes for the target class and supporting
13 classes; and
14 sharing the list of dependent classes with a second platform-independent
15 virtual machine so that the second platform-independent virtual machine does not
16 need to create the list of dependent classes.

1 11. (Original) The computer-readable storage medium of claim 10,
2 wherein the method further comprises, identifying classes that an object depends
3 upon by:
4 receiving a representation of the object;
5 serializing the referenced object;

6 parsing the data resulting from the object serialization to identify classes
7 referenced from the target object's properties, configuration, or state; and
8 if a target class is identified, determining the dependent classes of the
9 target class.

1 12. (Original) The computer-readable storage medium of claim 10,
2 wherein the method further comprises saving the list of dependent classes to a
3 storage structure.

1 13. (Original) The computer-readable storage medium of claim 12,
2 wherein the storage structure is one of a hash table and a database.

1 14. (Original) The computer-readable storage medium of claim 10,
2 wherein creating the list of dependent classes includes creating one of a
3 distribution list and a distribution file.

1 15. (Original) The computer-readable storage medium of claim 11,
2 wherein the method further comprises:
3 inserting the object into an object database;
4 determining if the target class and supporting classes for the target class
5 are in the class path; and
6 adding the target class and supporting classes for the target class to the
7 class path if necessary.

1 16. (Original) The computer-readable storage medium of claim 11,
2 wherein the method further comprises:
3 retrieving the object from the object database;

4 determining if the target class and supporting classes for the target class
5 are in the class path; and
6 adding the target class and supporting classes for the target class to the
7 class path if necessary.

1 17. (Original) The computer-readable storage medium of claim 10,
2 wherein the method further comprises filtering the list of identified classes to
3 remove duplicate and core class references.

1 18. (Original) The computer-readable storage medium of claim 10,
2 wherein the method further comprises saving the list of dependent classes of the
3 target class as well as the list of dependent classes of the supporting classes in
4 cache to facilitate subsequent lookups of dependent classes of the target class.

1 19. (Previously presented) An apparatus that determines a class
2 dependency that identifies a supporting class on which a target class depends,
3 wherein the target class is defined in an object-oriented programming language,
4 comprising:
5 a receiving mechanism that is configured to receive a representation of the
6 target class at a first platform-independent virtual machine;
7 a modeling mechanism that is configured to create a model of the target
8 class from the representation;
9 an analysis mechanism that is configured to analyze the model to detect
10 references to the supporting class;
11 a supporting mechanism that is configured to determine a class
12 dependency for the supporting class;
13 a listing mechanism that is configured to create a list of dependent classes
14 for the target class and supporting classes; and

15 a sharing mechanism that is configured to share the list of dependent
16 classes with a second platform-independent virtual machine so that the second
17 platform-independent virtual machine does not need to create the list of dependent
18 classes.

1 20. (Original) The apparatus of claim 19, wherein the receiving
2 mechanism is additionally configured to receive a representation of an object;
3 a serializing mechanism is configured to serialize the referenced object;
4 a parsing mechanism configured to parse the data resulting from the object
5 serialization to identify classes referenced from the target object's properties,
6 configuration, or state; and
7 a supporting mechanism that is configured to determine the dependent
8 classes of the target class.

1 21. (Original) The apparatus of claim 19, wherein the listing mechanism is
2 configured to save the list of dependent classes to a storage structure.

1 22. (Original) The apparatus of claim 21, wherein the storage structure is
2 one of a hash table and a database.

1 23. (Original) The apparatus of claim 19, wherein the listing mechanism is
2 configured to create the list of dependent classes, including creating one of a
3 distribution list and a distribution file.

1 24. (Original) The apparatus of claim 20, further comprising:
2 an insertion mechanism configured to insert the object into an object
3 database;

4 a determining mechanism configured to determine if the target class and
5 supporting classes for the target class are in the class path; and
6 an adding mechanism configured to add the target class and supporting
7 classes for the target class to the class path if necessary.

1 25. (Original) The apparatus of claim 20, further comprising:
2 a retrieving mechanism configured to retrieve the object from an object
3 database;
4 a determining mechanism configured to determine if the target class and
5 supporting classes for the target class are in the class path; and
6 an adding mechanism configured to add the target class and supporting
7 classes for the target class to the class path if necessary.

1 26. (Original) The apparatus of claim 19, further comprising a filtering
2 mechanism configured to filter the list of identified classes to remove duplicate
3 and core class references.

1 27. (Original) The apparatus of claim 19, further comprising a saving
2 mechanism configured to save the list of dependent classes of the target class as
3 well as the list of dependent classes of the supporting classes in cache to facilitate
4 subsequent lookups of dependent classes of the target class.